PERIODICA POLYTECHNICA SER. EL. ENG. VOL. 44, NO. 2, PP. 179–200 (2000)

COMPUTATIONAL ASPECTS FOR STEADY STATE ANALYSIS OF QBD PROCESSES

Hung T. TRAN and Tien V. DO

Department of Telecommunications
Budapest University of Technology and Economics
H–1117 Budapest, Pázmány Péter sétány 1/D, Hungary
e-mail: (hung,do)@plan.hit.bme.hu
Phone: 36 01 463 32 79

Received: Nov. 11, 2000

Abstract

In this paper, we give a survey on computational methods developed for steady state solution of QBD (Quasi Birth-Death) processes. Moreover, we adopt and implement a comparative framework to evaluate the capability of some chosen methods (spectral expansion, matrix geometric and its enhanced versions) in both finite and infinite cases. Numerical aspects concerning complexity, memory requirement and numerical stability are examined to expose the benefits and vulnerability of each method.

Keywords: QBD processes, steady state analysis, numerical methods, performance comparison.

1. Introduction

The QBD (Quasi Birth-Death) process was first introduced by WALLACE in [18] and EVANS in [9] in the late sixties. Basically, the QBD process is a two-dimensional (phase and level) Markov chain, where state changes are only allowed between ones having adjacent levels. The QBD process serves as a useful modelling tool in performance evaluation and system analysis, since it can be used to obtain solutions for several applied queueing models such as $M/PH/1/\infty$, $PH/M/n/\infty$, $PH/PH/1/\infty$, $MAP/PH/n/\infty$, $\sum_{i=1}^{n} MAP_i/PH/1/\infty$, $\sum_{i=1}^{n} MAP_i/PH/1/m$ ([12, 15, 16]).

Owing to its widespread applicability, the QBD process has gained a lot of research attention over recent years. For example, many research projects focus on finding numerical methods for steady state distribution of QBD processes. Moreover, evaluating the capability of those numerical methods also proves an important research issue and this is the main scope of this paper. Before moving ahead to detailed discussion, we now begin with the brief review of numerical methods available in the literature that were developed for steady state analysis of QBD processes.

The first numerical procedure known as a matrix geometric method was proposed by NEUTS in [16]. In this work, the geometric relation between level probability vectors was revealed, which makes the computation more convenient. The key element of this method is the iterative calculation of *rate matrix R*, by which the geometric relation is defined. However, the original matrix geometric method has some disadvantages mainly in terms of computational time. Therefore, improving the efficiency (e.g. time and space requirement, numerical stability) of this computational method is a great research challenge. In recent years, research efforts have resulted in several new computational methods published in the literature ([2, 3, 5, 7, 8, 13, 15, 20]).

The methods proposed by LATOUCHE et al. [13] and NAOUMOV et al. [15] are improved versions of the classical matrix geometric method. Having an in-depth analysis of QBD processes and using probabilistic interpretation, LATOUCHE et al. proposed a really fast and numerically stable algorithm for computing the rate matrix. The algorithm was speeded up by NAOUMOV et al. with the use of matrix factorization. These two methods are really popular and have been widely applied in several works.

Ram CHAKKA developed an exact computational method called *spectral expansion* for QBD processes [5, 6]. Instead of using the geometric relation between level probability vectors, a special expression of level probability vectors is introduced. The expression is defined by eigenvalues and eigenvectors of the characteristic matrix polynomial constructed from the process parameters. According to the author, this method is efficient, accurate and easy to use.

In [20], the authors presented an efficient and versatile *folding method*, that can be applied for finite QBD processes. The odd-even permutation achieved inside the transition probability matrix and the use of the principle of finite Markov chain reduction are the key elements of this computational method. In contrast with matrix geometric based methods, the folding algorithm solves directly the equation $\pi Q = 0$, where π is the steady state probability vector, Q is the generator matrix of the given QBD process. By taking a finite sequence of reduction steps (*forward reduction phase*), the original transition probability matrix is brought to a single -level form, from which a boundary vector can be determined. Since the steady state solution π is expressed as a product of the boundary vector and a finite sequence of expansion factors, it is calculable (*backward expansion phase*). Readers are encouraged to study [19, 20] for more details in the mathematical description as well as in the applicability of this method.

Nail AKAR et al. approach the solution of QBD processes from a novel side [2]. Their starting point is the observation of the close connection between solving the QBD process and solving the Algebric Ricatti Equations arising in optimal control problem in control theory [1]. Their proposed method then basically relies on the theory of invariant subspace and on the computation of matrix sign function with iterative procedure. The rate matrix R is obtained from a calculated invariant subspace of an adequately constructed matrix. The method is believed to be fast and stable.

In [3, 4], BINI and MEINI stated a fast, quadratically convergent and nu-

merically stable algorithm called *cyclic reduction algorithm* for QBD problems. The algorithm is derived from a block cyclic-reduction applied for block tridiagonal Toeplitz-like probability transition matrixes, supplemented with the use of FFT (*Fast Fourier Transform*) technique. Regarding the time complexity, this algorithm has been considered to be equivalent with Naoumov's algorithm.

Another solution for finite QBD processes was sketched in [8]. The authors provide an exact computational method, which is only based on simple matrix operations. The explicit analytic solution can be expressed in terms of process parameters. The authors show that the computation procedure has the same asymptotic complexity as other solving techniques. However, the applicability of their method is limited by the non-singularity condition of certain matrixes.

Recently, a novel method has been proposed in [7]. The method is named ETAQA (*Efficient Technique for the Analysis of QBD-processes by Aggregation*). In this method, the state space of a QBD chain is divided into several equivalent classes by a certain specific partitioning rule. Instead of computing the probability distribution of all states in the chain, only the aggregate probability distribution of the states in each class is evaluated. The authors show that those aggregate probabilities contain sufficient information to compute performance measures of interest such as the mean queue length or any higher moments. The method is proved to have appealing computational and storage complexity. ETAQA can be originally applied to a class of QBD processes, in which the downward transitions are directly towards a single state. If this condition is not fulfilled, further manipulations such as rearranging the state space partitioning are necessary.

In this paper, we perform a comparative study related to four computational methods developed for steady state analysis of QBD processes. Namely, the simple substitution matrix geometric, the logarithmic reduction algorithm of LATOUCHE et al., the algorithm proposed by NAOUMOV et al. and the spectral expansion method are included into the comparative study. Our preliminary goal is to give a comprehensive numerical comparison of the presently available methods and together with the results of comparative studies achieved in [5, 11, 14, 17] to construct a complete performance comparison picture. In this sense, both finite and infinite cases are considered in our work, whereas all the previous studies only deal with infinite cases. The criteria for comparison in our work include computational complexity (computational time), memory requirement, and numerical stability. Moreover, exploiting the fact that the spectral expansion method theoretically provides exact numerical results, we adopt and implement a comparative framework consisting of two kinds of stopping scenarios named object and performance parameter based scenarios for iterative methods.

The rest of this paper is organized as follows. Section 2 gives a brief description of QBD processes. In Section 3 and Section 4 numerical methods for infinite and finite QBD processes are presented, respectively. Section 5 presents the performance comparison between computational methods by delineating the comparative implementation as well as reporting numerical results. The case study taken into negotiation is a system of homogeneous processors with breakdowns and repairs. Section 6 ends the paper with summary and opened works.

2. Mathematical Description of QBD Processes

Consider a queueing system that can be modeled by a discrete time, two dimensional Markov process on semi-infinite or finite lattice strip. The process has a Markovian property and the state of system at observation time n can be described by two integer random variables I_n and J_n. The former one is bounded and referred to as a phase, the latter one may be either unbounded (infinite case) or bounded (finite case) and is referred to as a level of the system. The Markov process is denoted by $X = \{I_n, J_n; n \geq 0\}$ and its state space is $(\{0, 1, \ldots, N\} \times \{0, 1, \ldots\})$ in the infinite case and $(\{0, 1, \ldots, N\} \times \{0, 1, \ldots L\})$ in finite case, respectively. If the possible jumps of system's level in transition are only 0, -1 or 1, the corresponding process is known as Quasi Birth-Death (QBD) process.

The transition probabilities of the underlying Markov process are given by the following transition probability matrixes:

- A_j: purely phase transitions – From state (i, j) to state (k, j) $(0 \leq i, k \leq N; i \neq k; j = 0, 1, \ldots)$
- B_j: one-step upward transitions – From state (i, j) to state $(k, j+1)$ $(0 \leq i, k \leq N; j = 0, 1, \ldots)$
- C_j: one-step downward transitions – From state (i, j) to state $(k, j-1)$ $(0 \leq i, k \leq N; j = 0, 1, \ldots)$.

A_j, B_j and C_j matrixes have size of $(N+1) \times (N+1)$. We assume that for $j \geq M$ the transition matrixes become level-independent. That is

$$A_j = A,\ j \geq M; \quad B_j = B,\ j \geq M-1; \quad C_j = C,\ j \geq M. \tag{1}$$

For further computations we introduce the following notations:

- $p_{i,j}$: the steady state probability of the state (i, j)

$$p_{i,j} = \lim_{n\to\infty} P(I_n = i, J_n = j) \quad i = 0, 1, \ldots, N; \quad j = 0, 1, \ldots \tag{2}$$

  Our task is determining these probabilities in terms of known parameters of the system.
- $\underline{v}_j$: the row vector defined as:

$$\underline{v}_j = (p_{0,j},\ p_{1,j}, \ldots, p_{N,j}) \quad j = 0, 1, \ldots \tag{3}$$

- $\underline{e}$: the column vector of $(N+1)$ elements each of which is equal to 1.

For $j = 0, 1, \ldots, M-1$, the balance equations of the system are:

$$\underline{v}_j = \underline{v}_{j-1} B_{j-1} + \underline{v}_j A_j + \underline{v}_{j+1} C_{j+1}. \tag{4}$$

(It is assumed that $\underline{v}_{j-1} = \underline{0}$ if $j < 1$). For $j \geq M$, the corresponding j-independent set becomes the set of vector difference equations with constant coefficients:

$$\underline{v}_j = \underline{v}_{j-1} B + \underline{v}_j A + \underline{v}_{j+1} C, \quad j \geq M. \tag{5}$$

Note that if the system is finite ($j \leq L$) then one more boundary equation appears

$$\underline{v}_L = \underline{v}_{L-1} B + \underline{v}_L A. \tag{6}$$

In addition, since the sum of all probabilities must be one, we have:

$$\sum_{j=0}^{\infty} \underline{v}_j \underline{e} = 1.0 \tag{7}$$

for the infinite case and

$$\sum_{j=0}^{L} \underline{v}_j \underline{e} = 1.0 \tag{8}$$

for the finite case.

In order to get performance measures of the system, one has to know the steady state probabilities. The next sections will present some most well-known methods developed for steady state analysis of QBD processes. Although the algorithms are presented here for discrete time QBD processes, we emphasize that simple stochastic arguments approve their applicability in continuous time domain as well.

3. Computational Methods for Infinite QBD Processes

3.1. The Spectral Expansion Method

The main contribution of the so called spectral expansion method published by Ram CHAKKA [5, 6] is that the solution for *Eqs.* (5) can be expressed in the form

$$\underline{v}_j = \sum_{k=0}^{N} a_k \underline{\psi}_k \lambda_k^{j-(M-1)}, \qquad j \geq M-1, \tag{9}$$

where λ_k is the k-th eigenvalue strictly inside the unit disk and $\underline{\psi}_k$ is the corresponding left eigenvector of the characteristic matrix polynomial

$$Q(\lambda) = B + A\lambda + C\lambda^2,$$

i.e. they satisfy the equation

$$\lambda \underline{\psi}_k = \underline{\psi}_k Q(\lambda). \tag{10}$$

Combining the form (9) with the first M level-dependent *Eqs.* (4) and the normalized *Eq.* (7), one gets a set of linearly independent equations, which has a unique solution of $\underline{v}_0, \ldots, \underline{v}_{M-2}, \underline{a}$ where $\underline{a} = (a_0, \ldots, a_N)$ is the coefficient vector. The detailed procedure of computing all relevant eigenvalues and eigenvectors is discussed in [6] in an excellent way, therefore interested readers are referred to it for deeper insight.

3.2. The Matrix Geometric Method

In the classical matrix geometric method, NEUTS proved that the solution for *Eqs.* (5) is given as follows [16]:

$$\underline{v}_j = \underline{v}_{M-1} R^{j-(M-1)}, \qquad j \geq M-1, \tag{11}$$

where matrix R (referred to as the rate matrix) is the minimal non-negative solution of the quadratic matrix equation given by

$$B + RA + R^2 C = R. \tag{12}$$

Once R is determined, $\underline{v}_j$ $(j \geq M)$ can be expressed in terms of $\underline{v}_{M-1}$. Combining the form (11) with the first M level-dependent *Eqs.* (4) and the normalized *Eq.* (7), one gets a set of linearly independent equations, which has a uniform solution of $\underline{v}_0, \ldots, \underline{v}_{M-1}$.

To compute matrix R with the desired accuracy, several iterative procedures were offered in the last decade (see [13, 16]). The common feature of these methods is that all of them were formulated in terms of basic non-negative matrixes G, R, U, each of which has identical probabilistic interpretations as follows [13]

- $G(i, l)$ entry of matrix G is the probability that starting from state $(i, 1)$ the chain visits the level 0 and does so by visiting the state $(l, 0)$.
- $R(i, l)$ entry of matrix R is the expected number of visits into state $(l, 1)$ starting from state $(i, 0)$, until the first return to the level 0.
- $U(i, l)$ entry of matrix U is the taboo probability that starting from the state $(i, 1)$, the Markov chain eventually returns to the level 1 and does so by visiting the state $(l, 1)$, under taboo of the level 0 (i.e. without visiting any state in the level 0).

In the above definition of G, R, U matrixes, we may replace the levels 1 and 0 respectively by the levels $n + 1$ and n, for any $n \geq 0$.

The relation between the three matrixes is expressed by the following equations (see [10, 13])

$$G = (I - U)^{-1} C, \tag{13}$$
$$R = B(I - U)^{-1}, \tag{14}$$
$$U = A + BG = A + RC. \tag{15}$$

In addition, G, R, U matrixes are the minimal non-negative solutions of the non-linear equations

$$G = C + AG + BG^2, \tag{16}$$
$$R = B + RA + R^2 C, \tag{17}$$
$$U = A + B(I - U)^{-1} C. \tag{18}$$

Note, that once one of the three basic matrixes has been calculated, the other two are automatically obtained by means of *Eqs.* (13), (14), (15). Based on *Eqs.* (16), (17), (18), iterative procedures are proposed to compute G, R, U. In the case of positive recurrent QBD, the successive substitution procedure shown in *Fig. 1* can be used.

$k = 1$
$U = A$
$G = (I - U)^{-1}C$
DO
$\quad k = k + 1$
$\quad U = A + BG$
$\quad G = (I - U)^{-1}C$
$WHILE\ (||e - Ge||_\infty \geq \epsilon)$
$R = B(I - U)^{-1}$

Fig. 1. The iterative procedure of the matrix geometric method

This numerical algorithm has a complexity of $O(\frac{7}{3}(N+1)^3 I_U)$, where I_U is the necessary iterations needed to achieve a given accuracy ϵ.

3.3. The Logarithmic Reduction Algorithm by Latouche et al.

LATOUCHE and RAMASWAMI revealed in [13] the probabilistic interpretation hidden in the iterative procedure shown in *Fig. 1*. At each k-th iterative step, matrix G_k is evaluated. The element $G_k(i, l)$ is the probability that, starting from the state $(i, 1)$, the chain eventually visits the level 0, and does so by visiting the state $(l, 0)$, *under taboo* of the level $k + 1$ and above. In other words, in the k-th step, only those paths are considered, whose length does not exceed k. Thus, at the beginning of the algorithm, the chain is allowed to move no higher than the level 1. With each new iteration, the chain is allowed to visit one level above the previous maximum.

The main idea of the logarithmic reduction algorithm is derived from the stochastic observation mentioned above. In the new approach, during each iterative step, the chain is always allowed to proceed up to a multiple of twice the level attained at the previous iteration step. As a consequence, a logarithmic reduction in the number of iterations required to achieve convergence is performed. The iterative procedure is shown in *Fig. 2*.

The complexity of this algorithm is $O(\frac{25}{3}(N+1)^3 I_L)$, where I_L is the necessary iterations needed to achieve a given accuracy ϵ.

$T_0 = (I - A)^{-1}B$
$T_2 = (I - A)^{-1}C$
$k = 0$
$S = T_2$
$\Pi = T_0$
DO
$\quad k = k + 1$
$\quad T_i = (I - T_0T_2 - T_2T_0)^{-1}(T_i)^2 \quad i = 0, 2$
$\quad S = S + \Pi T_2$
$\quad \Pi = \Pi T_0$
$WHILE\ (||e - Se||_\infty \geq \epsilon)$
$G = S$
$U = A + BS$
$R = B(I - U)^{-1}$

Fig. 2. The logarithmic reduction algorithm by Latouche et al.

3.4. The Algorithm by Naoumov et al.

Based on the theory of matrix factorization, NAOUMOV et al. [15] developed further the logarithmic algorithm. Their computation algorithm reduces the complexity of the basic loop of each iteration step, herewith produces better performance. This improved iterative procedure is detailed in *Fig. 3*.

The complexity of this improved algorithm is $O(\frac{19}{3}(N+1)^3 I_N)$, where I_N is the necessary iterations needed to achieve a given accuracy ϵ.

4. Computational Methods for Finite QBD Processes

Imposing a limit on the maximum value of J_n leads to a QBD process having finite state-space $\{I_n, J_n; n \geq 0\}$. Let the maximum value of variable J_n be L, then the *Eqs. (5)* still hold, except that the range of j is limited to L.

4.1. The Spectral Expansion

Using spectral expansion implies the form [5]:

$$\underline{v}_j = \sum_{k=0}^{N} a_k \underline{\psi}_k \lambda_k^{j-(M-1)} + \sum_{k=0}^{N} b_k \underline{\phi}_k \beta_k^{L-j}, \qquad M - 1 \leq j \leq L. \tag{19}$$

$S = A - I$
$V = B$
$T = C$
$W = A - I$
DO
$X = -S^{-1}V$
$Y = -S^{-1}T$
$Z = VY$
$W = W + Z$
$S = S + Z + TX$
$V = VX$
$T = TY$
$WHILE\ (||Z||_\infty \geq \epsilon)$
$R = -BW^{-1}$

Fig. 3. The logarithmic reduction algorithm improved by Naoumov et al.

Here λ's are the $N+1$ eigenvalues of least absolute value determined from *Eq.* (10). β's are the $N+1$ eigenvalues of least absolute value satisfying equation

$$\underline{\phi}(C + A\beta + B\beta^2) = \beta\underline{\phi}. \tag{20}$$

$\underline{\psi}'$s and $\underline{\phi}'$s are eigenvectors corresponding to λ's and β's, respectively. Once the necessary eigenvalues and eigenvectors are determined, the set of linear simultaneous equations, which is composed of *Eqs.* (4), (6), (8), must be solved to determine $\underline{v}_0, \ldots, \underline{v}_{M-2}, \underline{a}, \underline{b}$, where $\underline{a} = (a_0, \ldots, a_N)$ and $\underline{b} = (b_0, \ldots, b_N)$.

4.2. Matrix Geometric, Latouche's and Naoumov's Methods

The matrix geometric solution is given by [2, 5]

$$\underline{v}_j = \underline{w}_1 R_1^{j-(M-1)} + \underline{w}_2 R_2^{L-j}, \qquad M-1 \leq j \leq L, \tag{21}$$

Here, $\underline{w}_1, \underline{w}_2$ are the unknown vectors of size $N+1$. R_1 and R_2 are the minimal non-negative solution of quadratic matrix equations

$$B + R_1 A + R_1^2 C = R_1, \tag{22}$$

$$C + R_2 A + R_2^2 B = R_2. \tag{23}$$

To compute R_1 and R_2 the matrix-geometric, Latouche's algorithm or Naoumov's algorithm can be used. Once those matrixes are calculated, the set of linear simultaneous equations, which is composed of *Eqs.* (4), (6), (8), must be solved to determine $\underline{v}_0, \ldots, \underline{v}_{M-2}, \underline{w}_1, \underline{w}_2$. For more detailed description, see [2, 5].

5. Numerical Comparison of Computational Methods for QBD Processes

5.1. Implementation

One observes that the computational procedure for obtaining the steady state probabilities of a QBD process consists of two phases[1]:

- In the first phase: either, all relevant eigenvalues and corresponding eigenvectors of the characteristic matrix polynomial are determined if the spectral expansion method is applied; or R matrix ($R1$ and $R2$ in case of finite process) is calculated if the MG, LA, NA are applied.
- In the second phase: a finite set of linearly independent equations must be solved for fundamental unknown probability vectors $\underline{v}_j$ $(0 \leq j \leq M-2)$ as well as for either additional vectors ($\underline{v}_{M-1}$ in the infinite case; $\underline{w}_1, \underline{w}_2$ in finite case) if MG, LA, NA is used; or coefficient vectors ($\underline{a}$ in infinite case; $\underline{a}, \underline{b}$ in finite case) if SE is used.

1. $k = 0,\ T_1 = 0$
2. *Run SE*
3. $Determine\ T_{SE}, E(j)_{SE}$
4. DO
   - 4.1. $k = k + 1$
     $T_2 = 0$
   - 4.2. *Run step k of METHOD*
     $t =$ *time of step k*
     $T_1 = T_1 + t$
   - 4.3. *Solve the set of equations*
     $T_2 =$ *Time of solving the set of equations*
   - 4.4. $Compute\ E(j)_{METHOD}$

   $WHILE\ (\frac{1}{E(j)_{SE}}(|E(j)_{SE} - E(j)_{METHOD}|) \geq \xi)$
5. *# of iterations* $= k$
6. $T_{METHOD} = T_1 + T_2$

Fig. 4. The iterative procedure applied in Scenario I for comparison

When using iterative methods for comparison, two scenarios are offered for stopping the iterative procedures.

- *Scenario I: Performance parameter based criteria*
  Theoretically, performance parameters such as the mean number of jobs in the system ($E(j)$) can be exactly calculated with spectral expansion due to its non-iterative feature. If one of them is adopted as *reference value* then

[1] As mentioned before, currently four methods have been implemented. They are Matrix Geometric (MG), Latouche's method (LA), Naoumov's method (NA) and the Spectral Expansion (SE)

the stopping criteria for the rest of iterative methods may be introduced as follows.

Let one of the performance measures (e.g. the mean queue length) calculated by matrix geometric method or by its improved versions be *comp. value*. The term *'relative bias'* or *'relative difference'* is defined as:

$$\text{relative bias} = \frac{|\text{reference value} - \text{comp. value}|}{\text{reference value}} \tag{24}$$

and is compared with the stopping criterion ξ in order to stop the iterative procedure shown in *Fig. 4* where METHOD term may be MG, LA or NA. In the infinite process, during one iterative step, matrix R is evaluated. In case of finite process the evaluation relates to both R_1 and R_2.

- *Scenario II: Object based criteria*
  The stopping criterion of the first scenario sometimes does not work, because the bias between the results belonging to the SE and other method cannot be reduced further, however many iterations are performed (see later). In these cases, we must use the original stopping criterion ϵ related to the object of the iterative procedure itself, which was described in Section 3.

The implementation of the methods was carried out in C using Meschach library[2] for matrix operation and solving a linear set of equations. All results reported are obtained by a program running on the Sun SPARCstation Ultra60 with sparc processor. The reported time is measured in seconds and is composed of the time of two computation phases.

5.2. Numerical Comparison

In what follows we compare the algorithms through a case study of the processor system with repairs and breakdowns [6]. The system is a set of homogeneous processors whose number is N. The processors break down from time to time. Single and independent failures of processors, as well as multiple and simultaneous failures are possible. The failed processor returns to operative state after successful repair. Single and independent repairs of processors, as well as multiple and simultaneous repairs are possible. Being in its operative state, each processor serves jobs, one at a time. Each job can occupy at most one operative processor at a time. Failure, repair and service time are assumed to be exponentially distributed. This system is modeled by a two dimensional, continuous Markov process in the following way:

- $I(t)$ is the operative state of the system, representing the number of operative processors at time t, $I(t) = 1, \ldots, N$,
- $J(t)$ is the number of jobs in the system at time t, $J(t) = 0, 1, \ldots$

[2]Meschach library for matrix operation is developed at the School of Mathematical Sciences, Australian National University by David E. Stewart and Zbigniew Leyk and it is free via netlib (ftp.netlib.org/c/meschach).

Now, we have to construct the transition matrixes A, B, C of the process, which have analogous interpretation with the discrete definitions given in Section 2. Let us notice that *when a new job arrives or when a completed job departs from the system, the operative state does not change, unless there is an independent coincidence towards such a change*. Hence, change in the operative state of the system is reflected only in the matrixes A and A_j. Taking into account that

- the individual processors break down independently at rate ξ and are repaired independently at rate η,
- the global simultaneous breakdowns of all currently operative processors occur at rate ξ_0 and the global simultaneous repairs of all currently inoperative processors occur at rate η_N

the A matrixes are given by:

$$A = A_j (j = 0, 1, \ldots) = \begin{bmatrix} 0 & N\eta & & & & \eta_N \\ \xi_0 + \xi & 0 & (N-1)\eta & & & \eta_N \\ \xi_0 & 2\xi & 0 & & & \eta_N \\ & \ddots & \ddots & \ddots & & \\ & & & & \eta + \eta_N & \\ \xi_0 & & & & N\xi & 0 \end{bmatrix} \tag{25}$$

When the operative state $I(t) = i$, jobs are assumed to arrive according to an independent Poison process with rate σ_i. For all j $(j = 0, 1, \ldots)$, the rate matrix of the one-step upward transitions (initiated by the arrivals of single jobs) has the form:

$$B = B_j = \text{diag}\,[\sigma_0, \sigma_1, \ldots, \sigma_N]. \tag{26}$$

Let us assume that each operative processor has an exponentially distributed service time with parameter μ. The one-step downward transitions take place by the departures of single jobs. The departure rate depends on the current state $(I(t), J(t)) = (i, j)$ and it is given by the entry $C_j(i, i)$ of matrix C_j. If $i > j$, then every job has a processor for getting service, and not all operative processors are occupied. Hence the departure rate of jobs is $j.\mu$. If $i \leq j$, then all the operative processors are occupied by jobs, hence the departure rate of jobs is $i.\mu$. We arrive at:

$$C_j = \text{diag}\,[0, \min(j, 1).\mu, \ldots, \min(j, N).\mu]. \tag{27}$$

Note that for $j \geq N$, C_j does not depend on j. Consequently, the threshold M is given by $M = N$.

In all the cases taken into account in this section, in order to keep the service capacity constant, the service rate of each processor was set to $\mu = 1$. For simplicity $\sigma_i = \sigma$ for all $0 \leq i \leq N$. Other parameters are $\eta = \eta_N = 0.1$ and $\xi = \xi_0 = 0.05$.

The criteria for comparison are

- computation complexity,
- numerical stability,

- and memory requirement.

The system is negotiated in both infinite and finite cases. The relevant characteristics of the system that affect the operation of the computational methods are the total average incoming load, the total average service rate (service capacity) and dimension of the system. These parameters are defined by means of parameters σ, μ, N and the transition matrixes.

In the infinite case, the total average incoming load should be less than the total average service rate, otherwise the system becomes unstable. In the finite case the system is always stable, but there will be lost events if the number of jobs exceeds the size of buffer (overload condition). For a system with finite buffer, our experiences show that the buffer's size has negligible impact on the computation time, therefore in all the cases reported below the buffer size was chosen $L = 100$.

5.2.1. Computational Complexity

First, we examine the dependency of computational time on the system load, stopping accuracy and the system's size. In the following tables, the terms I and T will refer to the number of needed iterations and the total computation time (measured in seconds), respectively.

Effect of system load. For the system of infinite buffer, the condition of stability is $\sigma < 0.666667 * N$. *Table 1* shows the number of the necessary iteration steps and computation time for a given relative bias ξ, while the load of the infinite system is increasing.

Table 1. Stability and complexity of computational methods in infinite QBD case

$\xi = 1e-3, N = 10$								
σ	Stability	T_{SE}	I_{MG}	T_{MG}	I_{NA}	T_{NA}	I_{LA}	T_{LA}
6	Y	0.03	629	0.27	9	0.03	8	0.03
6.6	Y	0.03	6633	3.00	12	0.03	11	0.03
6.66	Y	0.04	66461	29.73	16	0.03	15	0.03
6.666	Y	0.04	664736	274.89	19	0.03	18	0.04
6.6666	Y	0.04	6721786	2808.67	22	0.03	22	0.04
6.66666	Y	0.04	-	-	-	-	-	-
6.666666	Y	0.04	-	-	-	-	-	-
6.6666666	N	-	-	-	-	-	-	-

In *Table 1*, the term '-' is jotted down several times, even when the system was still stable. This is the case when the software package was unable to compute the rate matrix meeting a given relative bias either because the number of iterations is extremely high (MG case) or the fossilized result

calculated by LA and NA differs from the result of SE considerably. The first cause is illustrated in *Fig. 5*, where the needed iterations of MG increase with system load (through the arrival rate σ) in a exponential way.

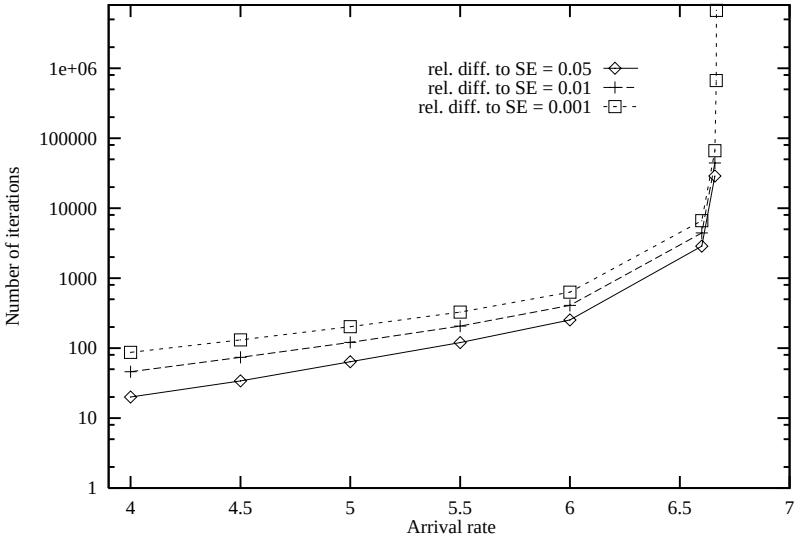


Fig. 5. Complexity of MG versus load of an infinite system ($N = 10$)

To expose the second cause, we have carried out further experiments as follows. We check the system on the verge of stability and choose the stopping criteria according to the Scenario II with $\epsilon = 1e - 12$. Numerical results are presented in *Table 2*.

Table 2. On the verge of stability of an infinite system ($N = 10$, $\epsilon = 1e - 12$)

σ (arrival rate)	I_{LA}	Rel. diff. of LA	I_{NA}	Rel. diff. of NA
6.66666	29	0.0069	28	0.0077
6.666664	30	0.0015	29	0.0012
6.666665	31	0.0383	30	0.0449
6.6666652	31	0.0209	30	0.0217
6.6666654	31	0.0154	30	0.0246
6.6666656	31	0.0343	30	0.0497
6.6666658	32	0.2473	31	0.2736
6.666666	33	0.3933	32	0.4013

As one can see, if the stability edge is approached very closely (remember that the service capacity of the studied system is $0.666667 * N$), the methods'

results show quite large bias and practically the relative bias $\xi = 0.001$ was never reachable. This explains why the computation procedure was unable to finish in case Scenario I was applied with relative bias $\xi = 0.001$.
For finite systems, the impact of the offered load on the computation time and number of necessary iterations can be seen in *Tables 3*, *4*.

Table 3. Computation time and number of iterations versus load σ (finite system, $\xi = 10^{-3}$, $N = 10$)

σ	T_{SE}	I_{MG}	T_{MG}	I_{NA}	T_{NA}	I_{LA}	T_{LA}
3.00	0.07	42	0.06	5	0.05	4	0.05
5.00	0.07	188	0.22	7	0.06	6	0.06
6.00	0.06	455	0.35	9	0.06	8	0.06
6.60	0.07	1230	0.56	10	0.06	9	0.06
6.66	0.07	1353	0.73	10	0.06	9	0.06
6.666	0.06	1357	0.78	10	0.06	9	0.06
6.6666	0.07	1358	0.87	10	0.06	9	0.06

For both cases of finite and infinite processes, numerical results presented in *Tables 1*, *3*, *4* clearly show that if the offered load is not heavy, the bias related with total computation time between SE, LA and NA is negligible. The computational time of MG tends to overstep the time of other methods if the offered load approaches the saturation condition (meanwhile the system is kept stable or is still able to operate without loss). Otherwise it operates nearly with the same efficiency.
Increasing the load (by increasing the arrival rate σ) has no significant effect on the computation time of spectral expansion, Latouche's and Naoumov's methods, i.e. their execution times are almost constant (see *Fig. 6*). The significant impact on the computation time of MG can be explained by slow convergence leading to a high number of iterations needed (as shown in *Fig. 5*).
It is interesting that in the finite case, for a given relative bias, the number of necessary iterations of MG method is not strictly increasing with the traffic load. After the saturation point, the number of iterations needed for a given relative bias tends to decrease which reduces the computation time in the way shown in *Fig. 6*.

Effect of stopping accuracy. If we claim more precise relative bias, then it takes longer to get numerical results because more iterations are needed. This tendency, however, is only considerable when MG is used. Generally speaking, the stricter the stopping criterion (related to the first scenario), the larger the complexity of the iterative procedure (see *Table 4*).
However, we just cannot assert that arbitrarily small relative bias is always reachable. The explanation for this is that after a certain number of iterations,

Table 4. Effect of the system load and the desired relative bias (ξ) on computation time (finite system)

$N = 10, \sigma = 3.0, E(j)_{\text{spect.exp}}$=5.19970358							
ξ	T_{SE}	I_{MG}	T_{MG}	I_{NA}	T_{NA}	I_{LA}	T_{LA}
1e-03	0.070	42	0.060	5	0.050	4	0.050
1e-06	0.070	107	0.130	7	0.060	6	0.060
1e-09	0.070	172	0.130	7	0.060	6	0.050
1e-12	0.070	236	0.280	8	0.060	7	0.060
$N = 10, \sigma = 6.0, E(j)_{\text{spect.exp}}$=33.55243677							
ξ	T_{SE}	I_{MG}	T_{MG}	I_{NA}	T_{NA}	I_{LA}	T_{LA}
1e-03	0.060	455	0.350	9	0.060	8	0.060
1e-06	0.060	1119	0.680	10	0.060	9	0.060
1e-09	0.070	1784	0.950	11	0.060	10	0.070
1e-12	0.070	2428	1.320	11	0.060	10	0.060
$N = 10, \sigma = 6.666, E(j)_{\text{spect.exp}}$= 51.0021848							
ξ	T_{SE}	I_{MG}	T_{MG}	I_{NA}	T_{NA}	I_{LA}	T_{LA}
1e-03	0.060	1357	0.780	10	0.060	9	0.060
1e-06	0.070	43377	24.780	15	0.070	14	0.070
1e-09	0.070	510683	278.350	19	0.070	18	0.060
1e-12	0.070	1176710	635.940	20	0.070	19	0.070

the performance measure produced by the iterative methods converges to a certain value. Letting the iterative procedure go on by making the stopping criteria of *Scenario I* smaller does not bring a significant change in this value, and so on the relative bias. This situation occurs mainly on the verge of stability border and is illustrated in *Fig. 7*. As we see, allowing longer run of the LA and NA iterative algorithms by decreasing the stopping criteria on ϵ does not mean better relative bias, it becomes unchanged.

Effect of the system's size. The capability of the applied methods in question can also be examined by changing the dimension of the system. *Table 5* shows the results for the infinite case that are obtained when N is changing. The parameter set is $\sigma = 0.6 * N$ (moderately loaded system), $\mu = 1.0$ and the relative bias $\xi = 1e-3$. *Table 6* reports the same investigation for the finite case $\sigma = 0.6 * N$ (moderately loaded system), $\mu = 10/N$ and the relative bias $\xi = 1e-3$.

One can observe that the total computation time increases with the system's size. In the case of infinite systems, the MG method seems a bit more time-consuming than the other ones. However, in the case of finite systems, all the four methods show almost the same efficiency related to computation time. The reasons for this may lie in two factors. First, the required relative bias

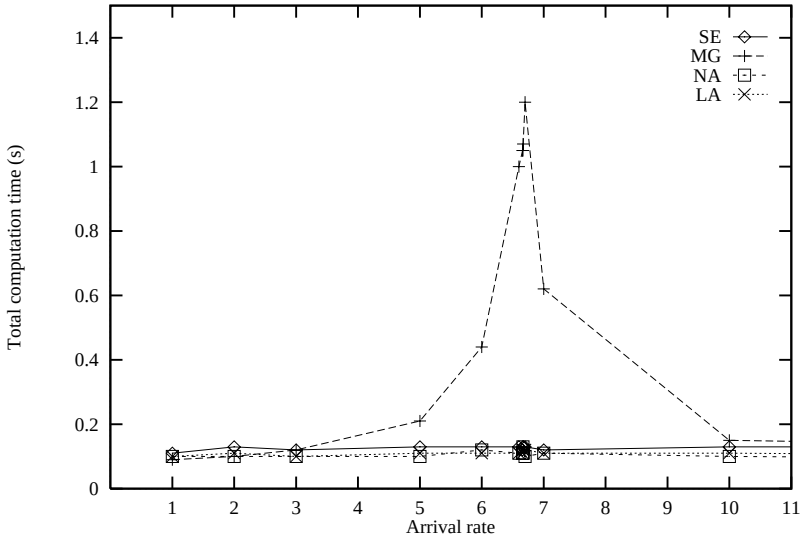


Fig. 6. Computation time versus load of a finite system ($N = 10$, $\xi = 10^{-3}$)

Table 5. Computation time versus system's dimension (infinite system, $\xi = 10^{-3}, \sigma = 0.6 * N$)

N	T_{SE}	I_{MG}	T_{MG}	I_{NA}	T_{NA}	I_{LA}	T_{LA}
5	0.0100	410	0.0600	8	0.010	7	0.0100
10	0.0400	629	0.3800	9	0.020	8	0.0300
15	0.2200	848	0.8800	9	0.200	8	0.2000
20	1.0500	1067	2.9200	10	1.030	9	1.0500
25	4.8000	1285	8.4200	10	4.750	9	4.7900
30	15.770	1504	22.300	10	15.680	9	15.680
35	42.780	1722	53.980	10	42.550	9	42.580
40	97.030	1941	114.61	11	96.770	10	96.810
45	202.38	2159	229.61	11	201.86	10	201.900
50	377.31	2377	420.58	11	376.9	9 10	377.080

is quite loose and secondly, the system is not considered in heavily loaded condition.

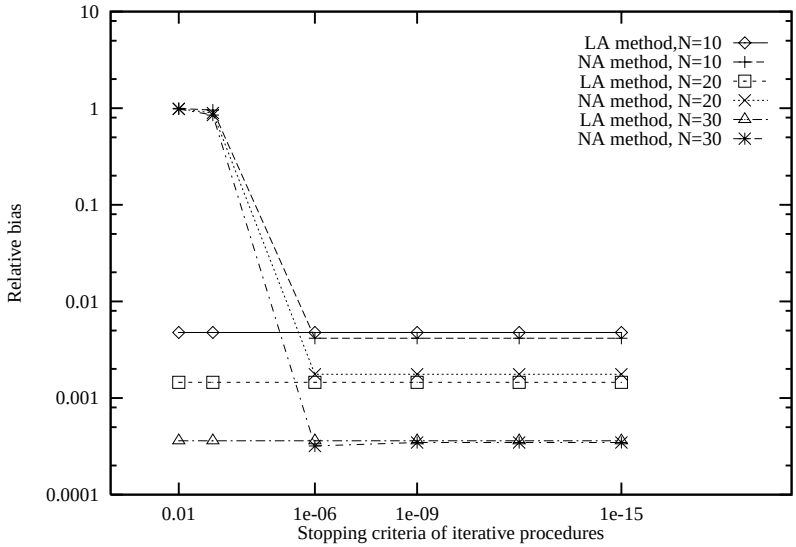


Fig. 7. Relative bias ξ versus stopping criteria ϵ ($\sigma = 0.666666 * N$)

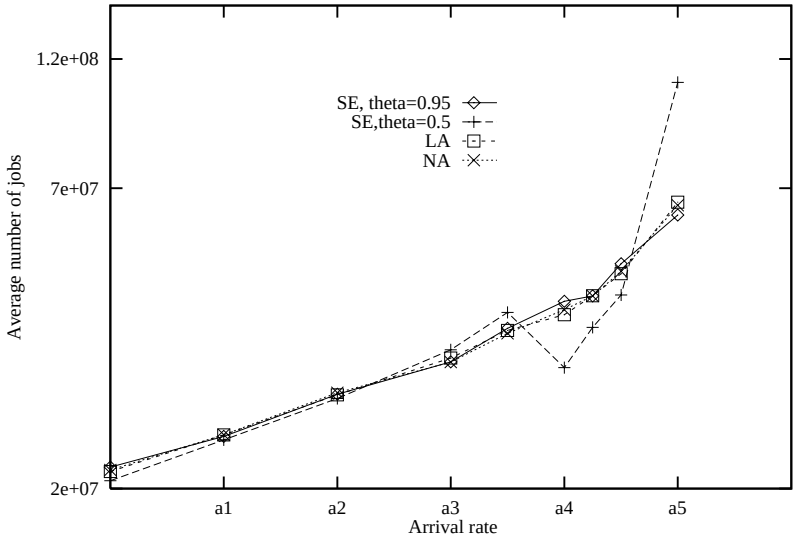
Table 6. Computation time versus system's dimension (finite system, $\xi = 10^{-3}, \sigma = 0.6 * N$)

N	T_{SE}	I_{MG}	T_{MG}	I_{NA}	T_{NA}	I_{LA}	T_{LA}
5	0.0200	13	0.0100	3	0.0100	2	0.010
10	0.0600	455	0.3500	9	0.0600	8	0.060
15	0.3700	112	0.4400	7	0.3400	6	0.340
20	1.3700	42	1.3500	5	1.3000	4	1.300
25	5.6100	26	5.4400	4	5.4100	3	5.410
30	17.880	19	17.550	4	17.5800	3	17.69
35	47.460	15	47.070	4	47.0700	3	47.08
40	107.08	12	106.56	4	106.430	3	106.34
45	217.29	11	216.64	3	216.770	2	216.62
50	405.62	10	404.67	3	404.700	2	404.73

5.2.2. Numerical Stability

Table 1 also confirms the stability of SE, LA and NA over MG. Differing from the MG case, the deterioration is not observed at the verge of the system's stability. However, if the considered system has a finite buffer, the computational methods do not show any failure during their operation. Experiences show that SE is sensitive

to the accuracy of computed eigenvalues. When a turning parameter (referred to as *theta* θ) should be used during the procedure calculating eigenvalues and eigenvectors (see [5]) and the system load is very close to the saturation point, some violations in numerical result occur. The situation is illustrated in *Fig. 8*. Taking a look at the neighborhood of $a4$, one can observe a failure exhibited in two facts. The first is that the results calculated with different θ-s are diverse. The second is that the fundamental rule, according to which the average number of jobs in the system must increase with the offered load, is violated.



	arrival rate (σ)
a1	6.6666652
a2	6.6666654
a3	6.6666656
a4	6.6666658
a5	6.666666

Fig. 8. On the verge of the system's stability

5.2.3. Memory Requirement

Let us turn now to the question of memory. *Table 7* shows the memory requirement needed during phase 1 of the computational procedure using each method. In the SE case, due to the computation algorithm of eigenvalues and eigenvectors discussed in [5], and based on the Schur decomposition implemented in the Meschach

library the maximum amount of necessary memory space is $11(N+1)^2+2(N+1)$ floating points when an infinite system is investigated. It is needed for storing (complex) eigenvalues and (complex) eigenvectors, as well as for allocating auxiliary matrixes. If the computation is for the finite system, then further more (complex) eigenvalues and (complex) eigenvectors must be stored. This fact leads to the need of $13(N+1)^2+4(N+1)$ floating points.

If MG is used, then for the iterative procedure, the auxiliary matrixes require $4(N+1)^2$ floating points. Since the R matrix must also be stored, the total memory requirement is $5(N+1)^2$. This memory amounts to $8(N+1)^2$ for LA and $10(N+1)^2$ for NA. When a finite system is investigated, instead of R, $R1$ and $R2$ matrixes must be stored. It means that $(N+1)^2$ more floating points are needed. If the Meschach library is left intact, then each matrix inverse operation that occurs in each iteration of MG, LA and NA, claims additional memory. Consequently if the number of iterations becomes extra large (MG case) then the program may have abnormal termination due to the lack of memory.

Table 7. Memory requirement

Method	Needed floating points (Infinite case)	Needed floating points (Finite case)
SE	$11(N+1)^2+2(N+1)$	$13(N+1)^2+4(N+1)$
MG	$5(N+1)^2$	$6(N+1)^2$
LA	$8(N+1)^2$	$9(N+1)^2$
NA	$10(N+1)^2$	$11(N+1)^2$

6. Conclusions

In this paper, we have dealt with the evaluation of numerical methods of a class of queueing models called Quasi Birth-Death process, that has a wide range of application in performance analysis of computer systems and telecommunications networks. We have reviewed the latest numerical methods for steady state analysis of QBD processes and subjected some of them to performance comparison. The comparative framework has been introduced by using two different stopping scenarios for iterative methods, and has been done for both infinite and finite cases. Important numerical aspects such as time complexity, numerical stability and memory requirement have been examined through a test example of a non-trivial repair-breakdown processor system.

Based on the discussion of numerical results, we have tried to give a thorough comparison of the performance of the computational methods. We have observed that a problem really arises when the QBD system approaches the stability bor-

der. When the system is slightly loaded, spectral expansion, matrix geometric, Latouche's and Naoumov's method are almost equally efficient, which does not hold for heavy load. In case of heavy load, it is not practical to use matrix geometric method because of its extremely long running time. Approaching closely the saturation point of the system, Latouche's and Naoumov's method may produce results with large difference compared to the result of spectral expansion due to some possible numerical problems of spectral expansion. Therefore, in a tight neighborhood of the saturation point it might not be advisable to use the spectral expansion method.

We are extending the comparative work in two directions. On the one hand, more practical examples should be involved as test systems. The more queueing phenomena are studied, the more overall and consistent comparative states can be concluded. On the other hand, it is desired to involve some other methods mentioned in the first section into the comparative study. At present, elaboration of both directions is under way.

References

[1] AKAR, N. – SOHRABY, K., System Theoretic Approach to Teletraffic Problems: A Unifying Framework, *Proceedings of GLOBECOM*, **1** (1996), pp. 163–167.

[2] AKAR, N. – SOHRABY, K., Finite and Infinite QBD Chains: A Simple and Unifying Algorithmic Approach, *Proceedings of IEEE INFOCOM*, (1997), pp. 1105–1113.

[3] BINI, D. – MEINI, B., On the Solution of a Nonlinear Matrix Equation Arising in Queueing Problems, *SIAM J. Matrix Anal. Appl.*, **17** (1996), pp. 906–926.

[4] BINI, D. – MEINI, B., Improved Cyclic Reduction for Solving Queueing Problems, *Numerical Algorithms*, **15** (1997), pp. 57–74.

[5] CHAKKA, R., Performance and Reliability Modelling of Computing System Using Spectral Expansion, University of Newcastle upon Tyne, PhD Thesis, 1995.

[6] CHAKKA, R. – MITRANI, I., A Numerical Solution Method for Multiprocessor Systems with General Breakdowns and Repairs, *Proceedings of 6th Int. Conf. on Performance Tools and Techniques*, pp. 289–304, September, 1992.

[7] CIARDO, G. – SMIRNI, E., ETAQA: An Efficient Technique for the Analysis of QBD-Processes by Aggregation, *Performance Evaluation*, **36–37** (1999), pp. 71–93.

[8] DE NITTO PERSONE, V. – GRASSI, V., Solution of Finite QBD Processes, *Applied Probability*, **33** (1996), pp. 1003–1010.

[9] EVANS, R. V., Geometric Distribution in some Two-Dimensional Queueing Systems, *Operations Research*, **15** (1967), pp. 830–846.

[10] HAJEK, B., Birth-and-Death Processes on the Integers with Phases and General Boundaries, *Journal of Applied Probability*, **19** (1982), pp. 488–499.

[11] HAVERKORT, B. – OST, A., Steady State Analysis of Infinite Stochastic Petri Nets: A Comparing between the Spectral Expansion and the Matrix Geometric Method, *Proceedings of the 7th International Workshop on Petri Nets and Performance Models*, pp. 335–346, 1997.

[12] KRIEGER, U. R. – NAOUMOV, V. – WAGNER, D., Analysis of a Finite FIFO Buffer in an Advanced Packet-Switched Network, *IEICE Trans. Commun.*, **E81-B** (1998), pp. 937–947.

[13] LATOUCHE, G. – RAMASWAMI, V., A Logarithmic Reduction Algorithm for Quasi-Birth-Death Processes, *Journal of Applied Probability*, **30** (1993), pp. 650–674.

[14] MITRANI, I. – CHAKKA, R., Spectral Expansion Solution of a Class of Markov Models: Application and Comparison with the Matrix-Geometric Method, *Performance Evaluation*, **23** (1995), pp. 241–260.

[15] NAOUMOV, V. – KRIEGER, U. R. – WARNER, D., Analysis of a Multi-Server Delay-Loss System With a General Markovian Arrival Process, In S. R. Chakravarthy and A. S. Alfa, editors, *Matrix-Analytic Methods in Stochastic Models*, Volume 183, *Lecture Note in Pure and Applied Mathematics*, Marcel Dekker, September, 1996.

[16] NEUTS, M. F., *Matrix Geometric Solutions in Stochastic Model*, Johns Hopkins University Press, Baltimore, 1981.

[17] OST, A. – HAVERKORT, B., Evaluating Computer-Communication Systems using Infinite-State Stochastic Petri Nets, *Proceedings of 3rd International Conference on Matrix Analytic Method*, pp. 295–314, 2000.

[18] WALLACE, V. L., *The Solution of Quasi Birth and Death Processes Arising from Multiple Access Computer Systems*, PhD Thesis, University of Michigan, 1969.

[19] YE, J. – LI, S. Q., Analysis of Multimedia Traffic Queues with Finite Buffer and Overload Control, Part II: Applications, *Proceedings of IEEE INFOCOM*, pp. 848–859, May 1992.

[20] YE, J. – LI, S. Q., Folding Algorithm: A Computational Method for Finite QBD Processes with Level-Dependent Transitions, *IEEE Trans. Commu.*, **42** pp. 625–639, February 1994.